On a conjecture of compatibility of multi-states characters

Michel Habib*, Thu-Hien To*

January 20, 2013

Abstract

Perfect phylogeny consisting of determining the compatibility of a set of characters is known to be NP-complete [5, 30]. We propose in this article a conjecture on the necessary and sufficient conditions of compatibility: Given a set $\mathcal{C}$ of r-states full characters, there exists a function $f(r)$ such that $\mathcal{C}$ is compatible iff every set of $f(r)$ characters of $\mathcal{C}$ is compatible. According to [8, 10, 9, 27, 12, 25], $f(2) = 2$, $f(3) = 3$ and $f(r) \geq r - 1$. [25] conjectured that $f(r) = r$ for any $r \geq 2$. In this paper, we present an example showing that $f(4) \geq 5$ and then a closure operation for chordal sandwich graphs. The later problem is a common approach of perfect phylogeny. This operation can be the first step to simplify the problem before solving some particular cases $f(4), f(5), \ldots$, and determining the function $f(r)$.

Keywords: perfect phylogeny, multi-states characters, chordal completion, triangulation

1 Introduction

Given an input biological data of a currently-living species set, phylogenetics aims to reconstruct evolutionary history of their ancestors. The evolutionary model of perfect phylogeny is phylogenetic tree, and the data are characters of species. Characters can be morphological, biochemical, physiological, behavioural, embryological, or genetic. Each character has several states. Here are some examples. The character *have wings* has two states: with wings and without wings. The character *number of legs* has many states: one leg, two legs, four legs, ... These are morphological characters. For an example of genetic characters, given a set of DNA sequences having a same length, if we consider each position on the sequences to be a character, then each character has 4 states corresponding to 4 bases of DNA as *A, T, C, G*.

Let $\mathcal{L}$ be a species set, and let c be a character on $\mathcal{L}$. Then, c can be represented by a partition of a non-empty subset $\mathcal{L}'$ of $\mathcal{L}$ such that each part consists of all species having the same state of c. So a set of characters is a set of partitions. A character is said to be trivial if the partition has at most one part having more than 1 element. Otherwise, it is non-trivial. If $\mathcal{L}' = \mathcal{L}$, then c is a full character, otherwise it is a partial character. If c has at most r parts, then c is a r-states character. A binary character is a 2-states full character.

Example 1 *Let $\mathcal{L} = \{a, b, c, d, e, f, g\}$ and $\mathcal{C} = \{c_1, c_2, c_3, c_4\}$ be a collection of characters on $\mathcal{L}$ such that:*

*Université Paris Diderot - Paris 7, LIAFA, Case 7014, 75205 Paris Cedex 13, France. E-mail: michel.habib/thu-hien.to/@liafa.jussieu.fr.

$c_1 = ab|cdefj|ghi$, $c_2 = def|abcghij$, $c_3 = gh|defi$, $c_4 = abcd|ghi$.

It means that c_1 *has* 3 *states* $0, 1, 2$ *such that the species* a, b *are in state* 0*, the species* c, d, e, f, j *are in state* 1*, and the species* g, h, i *are in state* 2*. Similarly for* c_2, c_3, c_4.

A phylogenetic tree on a species set $\mathcal{L}$ is a tree where each leaf is labelled distinctly by a species of $\mathcal{L}$.

Definition 1 *[7] Let* c *be a* r*-states character and let* T *be a phylogenetic tree on* $\mathcal{L}$*. For* $i = 0, \ldots, r-1$*, denote by* $T_i(c)$ *the minimal subtree of* T *on the leaf set consisting of the species having the state* i *of* c*. So,* c *is said to be* ***convex*** *on* T *iff the subtrees* $T_i(c)$ *are pairwise vertex-disjoint.*

A set of characters is **compatible** iff there exists a phylogenetic tree on which every character is convex. So the problem of perfect phylogeny is also commonly known as the character compatibility problem.

Example 2 *The set of characters* $\mathcal{C}$ *in Example 1 are compatible because there is a phylogenetic tree* T *in Figure 1 on which every character is convex. For example,* c_1 *is convex on this tree because the subtrees of* T *on* $\{a, b\}$*,* $\{c, d, e, f, j\}$ *and* $\{g, h, i\}$ *are pairwise vertex-disjoint. Similarly,* c_2*,* c_3*, and* c_4 *are also convex of this tree.*

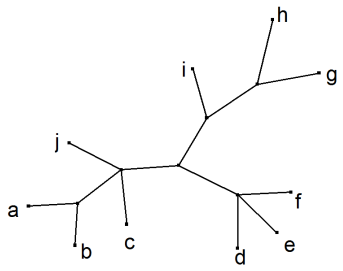


Figure 1: A phylogenetic tree on which all characters in Example 2 are convex

Determining the compatibility of a set of characters is NP-complete [5, 30]. In this article, we are interested in the necessary and sufficient conditions of compatibility of a set of r-states full characters. We propose the following conjecture.

Conjecture: For any set $\mathcal{C}$ of n r-states full characters, there exists a function $f(r)$, which does not depend on n, such that $\mathcal{C}$ is compatible iff every set of $f(r)$ characters of $\mathcal{C}$ is compatible.

This conjecture is based on the following previous results. According to [10, 27, 12], $f(2) = 2$ and according to [25], $f(3) = 3$. Fitch-Meacham examples [8, 9, 27, 25] showed that $f(r) \geq r$ for any $r \geq 2$. There are polynomial algorithms of checking compatibility of 3-states full characters [7] and 4-states full characters [23]. By [26], if the number of states is restricted, then checking the compatibility of $\mathcal{C}$ is polynomial. In general, [1, 24, 2] showed that there is a polynomial algorithm in the number of characters and species, but exponential in the number of states.

Some related work

Existence of perfect phylogeny: Given a set of characters on $\mathcal{L}$, is there any phylogenetic tree on which all characters are convex? It is easy to determine whether a collection of binary characters is compatible. However the problem is NP-complete even for 2-states characters [5, 30]. There exists effective, practical approaches for 2-states characters [19], and a new approach for this problem is proposed in [14].

Quartet problem: A minimal non-trivial character is a 2-states character such that each state contains exactly two species. Such character is called a *quartet*. As stated in the previous paragraph, the problem of compatibility of a set of quartets is NP-complete [5, 30]. However, there are some particular cases that the problem is polynomial [3], see [28] for details.

Define a tree by characters: a set of characters define a tree iff there is not any other tree on which these characters are convex. [29] showed that a set of 3 characters are not sufficient to define a tree but a set of 5 characters are. Later, [22] showed that for any tree, there exist at most 4 characters which define this tree. Hence, 4 is the optimal value. For the problem of whether a set of characters defines a tree, this is recently proved to be NP-hard [20]. The author solved the quartet challenge proposed by [28] as follows: given a phylogenetic tree on $\mathcal{L}$, and a quartet set $\mathcal{Q}$ which is convex on this tree. Is there any other tree on which this quartet set is also convex? Despite the NP-hardness, there is a polynomial algorithm for this problem when $|\mathcal{L}| - |\mathcal{Q}| = 3$ [3, 4].

Maximum parsimony: When there is no perfect phylogeny that can be inferred from data, it is desirable to find a model that minimize the number of reverse and convergent transitions. That is the problem of maximum parsimony.

Perfect phylogeny with recombination: When the characters set are not tree-representable, it is also interesting to construct a model that can represent phylogenies. The model used here is recombination networks. Introduced by [21], then intensive work have been done on this problem, including [31, 16, 18, 17, 13, 15].

2 Preliminaries

A very popular approach of perfect phylogeny is using chordal completion of vertex-coloured graphs, or equivalently chordal sandwich graph.

Definition 2 *Let $\mathcal{L} = \{x_1, \ldots, x_m\}$ be a species set and let $\mathcal{C} = \{c_1, \ldots, c_m\}$ be a set of characters on $\mathcal{L}$. Each c_i is a partition of a subset of $\mathcal{L}$. The* ***partition intersection graph*** *$G = (V, E)$ of $\mathcal{C}$ is constructed as follows:*

- Each character of $\mathcal{C}$ is associated with a different colour.

- Each vertex of V corresponds to a state of a character of $\mathcal{C}$. This vertex is then coloured by the colour of the character.

- There is an edge between 2 vertices if the 2 corresponding states of the 2 characters have at least a common species.

In our figures, in stead of colouring the vertices, we include the name of the characters in the labels of the vertices.

Example 3 *Let consider the character set in Example 2: $\mathcal{C} = \{c_1, c_2, c_3, c_4\}$ where*

$c_1 = ab|cdefj|ghi = c_{1,0}|c_{1,1}|c_{1,2}$,

$c_2 = def|abcghij = c_{2,0}|c_{2,1}$,
$c_3 = gh|defi = c_{3,0}|c_{3,1}$,
$c_4 = abcd|ghi = c_{4_0}|c_{4,1}$.

Each vertex $c_{i,j}$ represents the state j of character i. The partition intersection graph of $\mathcal{C}$ is in Figure 2.

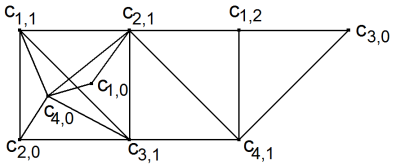


Figure 2: An example of partition intersection graph

A graph G is *chordal* if every cycle of length ≥ 4 contains at least a chord. A *chordal completion* of G is a chordal graph $G' = (V, E')$ such that $E \subseteq E'$. This completion is minimal iff when we remove any edge in $E' \setminus E$, the resulting graph is not chordal.

Given a vertex-coloured graph G, a **proper chordal completion** of G is a chordal graph $G' = (V, E')$ such that $E \subseteq E'$ and E' does not contain any edge connecting two vertices of the same colour.

Theorem 1 *[6, 27, 30] A set of characters $\mathcal{C}$ is compatible iff its partition intersection graph has at least a proper chordal completion.*

The set of characters in Example 3 is compatible as showed in Example 2. Its partition intersection graph (Figure 2) has indeed a proper choral completion which is itself.

Proper chordal completion of vertex-coloured graph can be stated equivalently under the form of sandwich problems.

Definition 3 *Given a graph $G = (V, E, F)$ where F is a set of pairs of vertices of G such that $E \cap F = \emptyset$.*

If there is a graph $G_S = (V, E_S)$ such that $E \subseteq E_S \subseteq E \times E \setminus F$ and G_S satisfies property Π, then G_S is called a Π***-sandwich graph*** *of G.*

See [11] for some problems and results on graph sandwich problems.

It is easy to see that a chordal completion of a vertex-coloured graph $G = (V, E)$ is proper iff it is a chordal-sandwich graph of $G = (V, E, F)$ where F is the set of pairs of vertices having a same colour. So, by considering the set F, we can ignore the colours of the initial graph. We also call a *chordal-sandwich graph* of G a *proper chordal completion* of G, i.e. a chordal completion of G without using any pair of vertices in F.

3 Our contributions

Fitch-Meacham examples were first introduced in [8, 9], then later generalized in [27] and formally proved in [25], showed that $f(r) \geq r$ for any $r \geq 2$. [25] conjectured that for any r, there is a

perfect phylogeny on r-state characters if and only if there is one for every subset of r characters, i.e. $f(r) = r$ for any $r \geq 2$. However, we have an example in Section 4 showed that $f(4) \geq 5$. It improves the lower bound of Fitch-Meacham examples and shows that the conjecture in [25] is not true. After that, in Section 5, we propose a closure chordal sandwich graph operation such that the obtained graph has a stronger structure. As a consequence, one can suppose that the input graph has such a structure, which can facilitate settling the conjecture.

4 An example of 4-States Characters

We present here an example of a set of 4-states characters which is not compatible, but every 4 characters of this set are compatible.

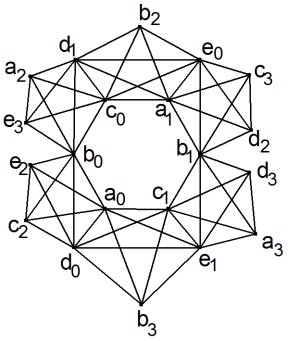


Figure 3: Graph G

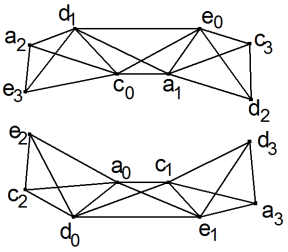


Figure 4: The induced subgraph of G on 4 colours a, c, d, e. This graph is chordal.

Let $\mathcal{C}$ be the following set of characters:
$a = \{x, u\}|\{z, t\}|\{y\}|\{v\} = a_0|a_1|a_2|a_3$
$b = \{x, y\}|\{t, v\}|\{z\}|\{u\} = b_0|b_1|b_2|b_3$
$c = \{y, z\}|\{u, v\}|\{x\}|\{t\} = c_0|c_1|c_2|c_3$
$d = \{x, u\}|\{y, z\}|\{t\}|\{v\} = d_0|d_1|d_2|d_3$
$e = \{z, t\}|\{u, v\}|\{x\}|\{y\} = e_0|e_1|e_2|e_3$

Each character has 4 states that we denote for example by a_0, a_1, a_2, a_3 for the character a. The partition intersection graph G associated to $\mathcal{C}$ is in Figure 3.

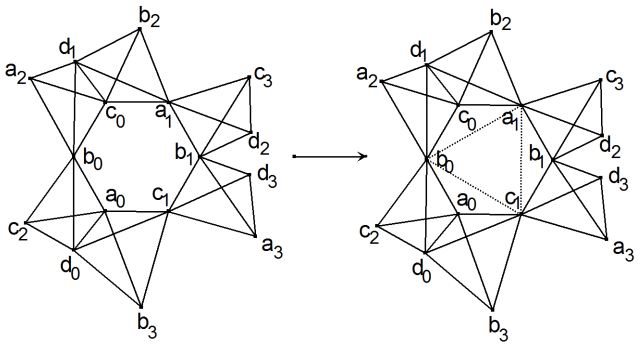


Figure 5: The induced subgraph of G on 4 colours a, b, c, d and its chordal completion

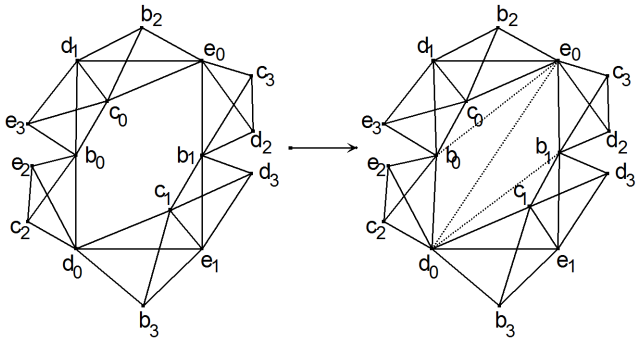


Figure 6: The induced subgraph of G on 4 colours b, c, d, e and its chordal completion

G does not accept any proper chordal completion. Indeed, if we consider only the induced subgraph of G on 4 colours a, b, c, d and triangulate it, then there is a unique way to do that by connecting $(a_1, b_0), (b_0, c_1)$ and (c_1, a_1) (Figure 5). Similarly, if we consider the induced subgraph of G on 4 colours a, b, c, e, there is also a unique way to triangulate it by connecting $(a_0, b_1), (b_1, c_0)$ and (c_0, a_0). So, to triangulate G, the cycle $(a_0b_1a_1b_0, a_0)$ is forced to be created. However, this cycle does not have any proper chordal completion. In other words, G does not have any proper chordal completion. So, $\mathcal{C}$ is not compatible.

However, as we see in Figure 4, the induced subgraph of G on 4 colours a, c, d, e is chordal. The induced subgraph of G on 4 colours a, b, c, d has a proper chordal completion (Figure 5) and similarly for the induced subgraph of G on 4 colours a, b, c, e due to the symmetry. The induced subgraph of G on 4 colours b, c, d, e also has a proper chordal completion (Figure 6) and similarly for the induced subgraph of G on 4 colours a, b, d, e.

It means that every 4 characters of $\mathcal{C}$ are compatible but the whole set $\mathcal{C}$ is not compatible.

5 A closure chordal sandwich graph operation

Given a graph $G = (V, E, F)$ where $E \cap F = \emptyset$. Let u, v be two vertices of G such that $(u, v) \notin E$.

(u, v) is a *forbidden edge* if it is not included in any minimal proper chordal completion of G. So (u, v) is forbidden if either $(u, v) \in F$ or if by connecting them, the resulting graph does not have any proper chordal completion. The forbidden edges are presented by dotted lines in our figures.

(u, v) is a *forced edge* if it is contained in every proper chordal completion of G. So if there is a cycle in G which has a unique proper chordal completion, then the edges used to complete this cycle are forced.

Note that by adding any forced edge into E or any forbidden edge into F, we do not lose any proper chordal completion.

A cycle C of G is forbidden if every chordal completion of C contains at least an edge in F. So if G has a proper chordal completion then it does not have any forbidden cycle. The converse is not always true. For example, see the graph $G = (V, E, F)$ in Figure 8 where F consists of the pairs of vertices having a same colour. This graph has 3 chordless cycles and each one can be chordally completed without using any edges in F. However, G does not admit any proper chordal completion.

Example 4 *In Figure 7 we have a cycle of size* 5 *on* 3 *colours* a, b, c. *The set* F *consists of* (a_0, a_1) *and* (b_0, b_1). *One can deduce that* (a_1, b_1) *is forbidden since by connecting them we have the forbidden cycle* $(a_0b_0a_1b_1)$. *We deduce furthermore that* (c_0, a_0) *and* (c_0, b_0) *are forced because the unique way to properly chordally complete this cycle is connecting them.*

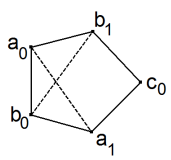


Figure 7: $(a_0, a_1), (b_0, b_1), (a_1, b_1)$ are forbidden edges. (c_0, a_0), (c_0, b_0) are forced edges.

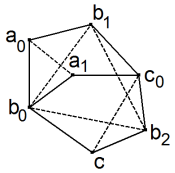


Figure 8: A graph which does not have any proper chordal completion

Consider the graph in Figure 8 where F is the set of pairs of vertices having a same colour. Similarly to the previous example on the cycle $a_0b_1c_0a_1b_0$, we deduce that c_0a_0 and c_0b_0 are forced. So, the cycle $b_0c_0b_2c_1$ is forced to be presented in any proper chordal completion of this graph. However, this cycle is forbidden. Therefore, this graph does not have any proper chordal completion.

Observation 1 *Given a cycle $C = (u_1 \dots u_k, u_1)$, then:*

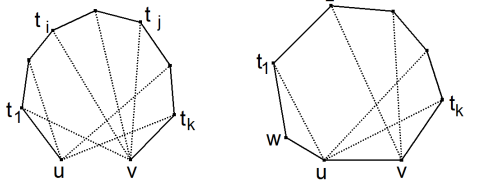
(i) for any $i \in \{1, \dots, k\}$, every chordal completion of C must contain either (u_{i-1}, u_{i+1}) or (u_i, u_j) for a certain j different from $i, i-1, i+1$.

(ii) every chordal completion of C must contain a chord (u_{i-1}, u_{i+1}) for a certain i.

Lemma 1 (Detecting forbidden edges and forced edges) *Let $G = (V, E, F)$ be a graph where $E \cap F = \emptyset$ and (u, v) be two vertices of G:*

1) If there is a chordless path $(ut_1 \dots t_kv)$ such that for any $i = 1, \dots k$, either (u, t_i) or (v, t_i) is forbidden, then (u, v) is also forbidden.

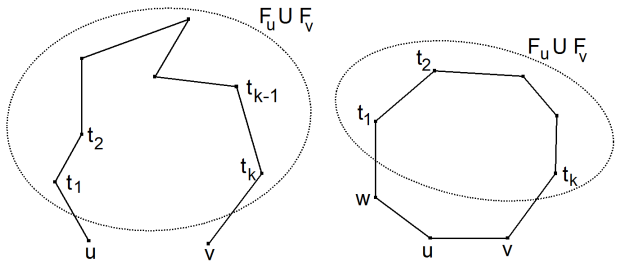
2) Suppose that $(u, v) \in E$. If there is a chordless cycle $c = (uwt_1 \dots t_kv, u)$ such that for any $i = 1, \dots k$, either (u, t_i) or (v, t_i) is forbidden, then (v, w) is a forced edge.



(a) (u, v) is forbidden

(b) (v, w) is a forced edge

Figure 9: Lemma 1



(a) a $f(u, v)$ path, (u, v) is forbidden.

(b) a $g(u, v, w)$ cycle. (w, v) is forced.

Figure 10: Corollary 1

Proof: 1) By connecting (u, v), we obtain the chordless cycle $C = (ut_1 \dots t_kv, u)$. We will prove the cycle C is forbidden. By Observation 1 (i), to chordally complete C, we must connect either (t_1, v) or (u, t_i) for a certain $i = 2, \dots, k$. However, (t_1, v) is forbidden because (u, t_1) is an edge of G and by the assumption, either (u, t_1) or (v, t_1) must be forbidden. So, we must connect an edge (u, t_i), which must not be a forbidden edge. We deduce that (v, t_i) is forbidden. The created subcycle $(uu_iu_{i+1} \dots u_kv, u)$ has the same property as C. So, by using the same argument, to chordally complete this cycle, we must connect an edge (u, u_j) where $i < j \leq k$. So, the size of the considering cycle decreases strictly each time applying this argument. Then there will be a moment that we obtain a cycle which is forbidden, in other words the cycle C is forbidden.

2) We will prove that, every proper chordal completion of C must contain the chord (v, w). Suppose that there is a proper chordal completion of this cycle which does not contain (v, w). By Observation 1 (i), this completion must contain (u, t_i) for a certain $i = 1, \dots, k$. We obtain then a subcycle $(ut_it_{i+1} \dots t_kv, u)$. The path $(ut_it_{i+1} \dots t_kv)$ satisfies the condition of Claim 1, so (u, v) is a forbidden edge. However, (u, v) is an edge of G. It means that this subcycle does not have any proper chordal completion. In other words, C does not have any proper chordal completion which does not contain (u, v). So (v, w) is presented in any chordal completion of G, i.e. it is a forced edge. □

Corollary 1 *Let a graph $G = (V, E, F)$, denote by $F(u)$ the set of vertices u' such that $(u, u') \in F$. Then, for any pair of vertices (u, v):*

1) If (u, v) is not an edge of G and there is a chordless path $(ut_1 \dots t_k v)$ such that for any $i = 1, \dots, k$, $t_i \in F(u) \cup F(v)$ then (u, v) is forbidden. We call such a path a $f(u, v)$ path.

2) If (u, v) is an edge of G and there is a chordless cycle $(uwt_1 \dots t_k v, u)$ such that $w \notin F(v)$ and for any $i = 1, \dots, k$, $t_i \in F_u \cup F_v$, then (v, w) is a forced edge. We call such a cycle a $g(u, v, w)$ cycle.

Denote by $N(u)$ the set of neighbour vertices of u.

```
   Data: A graph G = (V, E, F)
   Result: Closure(G)
 1 For any u ∈ V, calculate N(u) and F(u) = {v| (u, v) ∈ F};
 2 flag = true;
 3 while (flag) do
 4 |   flag = false;
 5 |   for (any pair of vertices (u, v)) do
 6 |   |   if (there is a f(u, v) path) then
 7 |   |   |   if (u ∉ N(v)) then
 8 |   |   |   |   Add u to F(v), and v to F(u);
 9 |   |   |   |   flag=true;
10 |   |   |   else
11 |   |   |   |   G does not have any proper chordal completion. Exit;
12 |   |   if (u ∈ N(v)) ∧ (there is a g(u, v, w) cycle) then
13 |   |   |   Add v to N(w), and w to N(v);
14 |   |   |   flag=true;
15 return G' = (V, E', F') where E' = {(u, v)| u ∈ N(v)} and F' = {(u, v)| u ∈ F(v)}.
```

Algorithm 1: A closure chordal-sandwich graph operation

Theorem 2 *Algorithm 1 takes time $O(n^4(n + m))$. Let $G' = (V, E', F') = Closure(G)$, then any proper chordal completion of G' is a proper chordal completion of G and vice-versa. Moreover, G' satisfies the following properties:*

1. For any $(u, v) \notin E' \cup F'$, if we connect (u, v) then for any created chordless cycle C which has (u, v) as an edge, C has at least a chordal completion without using any pair of vertices in F'.

2. Any chordless cycle of G' has at least two chordal completions without using any pair of vertices in F'.

Proof: **Correctness:**

According to Corollary 1, the loop *for* recognizes all pairs of vertices (u, v) which satisfy the conditions in the corollary to detect the forbidden edges and forced edges. So the pairs of vertices added in F' at line 8 are the forbidden edges, i.e. the edges which are not included in any proper chordal completion of G. And the pairs of vertices added in E' at line 13 are forced edges, i.e. the edges which are included in every proper chordal completion of G. Therefore, any proper chordal completion of the obtained graph is a proper chordal completion of G and vice versa.

Moreover, if (u, v) is an edge and there exists a $f(u, v)$ path then according to the proof of Lemma 1, the chordless cycle consisting of $f(u, v)$ and (u, v) is forbidden. Hence, G does not have any proper chordal completion (line 11). This process stops when there is no more forbidden edges or forced edges detected. So, in G', for any pair of vertices (u, v), there is no $f(u, v)$ path; and if $(u, v) \in E'$, then there is no $g(u, v, w)$ cycle. We will prove that G' satisfies the two properties of the theorem.

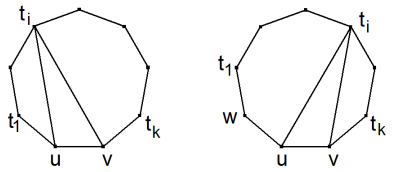


Figure 11: Proof of Theorem 2

1) The first property: we prove by induction on the size of cycles.

Let (u, v) be a pair of vertices not in $E' \cup F'$. By connecting (u, v), let C be a created cycle which contains (u, v). We will prove that C admits at least a chordal completion without using any edge in F'.

For the case $|C| = 4$, let $C = (uxyv, u)$. By the assumption, G' does not contain any $f(u, v)$ path, i.e. $(uxyv)$ is not a $f(u, v)$ path. It means that either $(v, x) \notin F'$ or $(u, y) \notin F'$. So, we can complete C by connecting either (v, x) or (u, y).

Suppose that C has at least a chordal completion if $|C| \leq k$.

For the case $|C| = k+1$, let $C = (ut_1 \dots t_{k-1}v, u)$, so there is not any $f(u, v)$ path in G'. Then, there exists at least an $i \in \{1, \dots, k-1\}$ such that $(u, t_i), (v, t_i) \notin F'$. By the induction hypothesis, if we connect (u, t_i), (v, t_i), the two subcycles $(ut_1 \dots t_i, u)$ and $(vt_i \dots t_k, v)$ have proper chordal completions without using any pair of vertices in F' because both these two cycles have size smaller than $k + 1$ (Figure 11(a)). Completing these two subcycles gives a proper chordal completion for C. So, C admits at least one chordal completion, the one which contains (u, t_i) and (v, t_i).

2) The second property: Suppose that there is a chordless cycle C which admits a chordal completion without connecting any pair of vertices in F'. By Observation 1 (ii), this chordal completion must contain at least a triangle (u, v, w) such that (u, v), (u, w) are edges of C. Let $C = (uwt_1 \dots t_k v, u)$, so C is not a $g(u, v, w)$ cycle because otherwise (v, w) is a forced edge and it must have been connected by the algorithm, a contradiction with the fact that C is chordless. So, there is a t_i such that $(u, t_i), (v, t_i) \notin F'$ (Figure 11(b)). Using the first property, if we connect (u, t_i) and (v, t_i), then we obtain two subcycles which have proper chordal completions without connecting any pair of vertices in F'. That implies another chordal completion of C containing (u, t_i) and (v, t_i). This chordal completion does not contain (v, w), so it is different with the initial one. In other words, C has at least 2 distinct proper chordal completions without using any pair of vertices in F'.

Complexity:

- Calculating $N(u)$ and $F(u)$ for any vertex u in line 1 is done in times $O(n^2)$.

- The loop *while*: For each iteration, there is at least a pair of vertices (u, v) whose nature is modified, i.e (u, v) becomes either a forbidden edge or a forced edge. Once it is modified, it will not be modified afterwards. The loop stops when there is no more modification on any pair of vertices. So, the number of iterations of this loop is bounded by the number of pairs of vertices, i.e by $O(n^2)$.

- The loop *for*: there are $O(n^2)$ pairs of vertices (u, v). So there are $O(n^2)$ iterations. In each iteration:

• Checking if there is a simple path $f(u, v)$ can be done in linear time: We proceed a dfs starting at u such that the visited vertices are in $F(u) \cup F(v) \setminus N(u)$. If we meet a vertex in $N(v)$, then there is a $f(u, v)$ path. Otherwise, there is no such path.

• Checking if there is a $g(u, v, w)$ cycle can also be done in linear time: We proceed a dfs from u such that the first visited vertices is not in $N(v) \cup F(v)$, and the remaining visited vertices are in $F(u) \cup F(v) \setminus N(u)$. If we meet a vertex in $N(v)$ then we have a $g(u, v, w)$ cycle. Otherwise, there is no such cycle.

So, the total complexity is $O(n^4(n + m))$ where n is the number of vertices of G' and m is the number of edges of the obtained graph. □

With this operation, one can deduce for example that the input graph does not contain any cycle as in Figure 7 because this cycle must be already triangulate by the operation. Or the input graph can not contain an induced subgraph as in Figure 8, because the operation can show that in this case the graph does not have any proper chordal completion.

Corollary 2 *Without loss of generality, one can suppose that any cycle of a graph $G = (V, E, F)$ has at least two proper chordal completions; and by connecting any pair of vertices not in $E \cup F$, every created cycle has at least one proper chordal completion.*

6 Conclusion

Our example in Section 4 showed that $f(4) \geq 5$. We suggest that $f(r) \geq r + 1$ for any $r \geq 4$. So, a further work is to generalize our example, or to find other examples supporting this suggestion. Another problem is proving that $f(r)$ exists by determining an upper bound function of r for $f(r)$. It means that we must find a function $F(r)$ such that if every set $F(r)$ characters of $\mathcal{C}$ is compatible then $\mathcal{C}$ is compatible. A harder question is determining $f(r)$ for $r \geq 4$. Our closure operation for chordal sandwich graphs can help to simplify these problems.

References

[1] Richa Agarwala and David Fernández-Baca. A polynomial-time algorithm for the perfect phylogeny problem when the number of character states is fixed. *SIAM J. Comput.*, 23(6), 1994.

[2] Richa Agarwala and David Fernández-Baca. Simple algorithms for perfect phylogeny and triangulating colored graphs. *Int. J. Found. Comput. Sci.*, 7(1):11–22, 1996.

[3] Sebastian Bcker, Andreas W.M. Dress, and Mike A. Steel. Patching up x-trees. *Annals of Combinatorics*, 3:1–12, 1999.

[4] Sebastian Böcker, David Bryant, Andreas W. M. Dress, and Mike A. Steel. Algorithmic aspects of tree amalgamation. *J. Algorithms*, 37(2):522–537, 2000.

[5] Hans L. Bodlaender, Michael R. Fellows, and Tandy Warnow. Two strikes against perfect phylogeny. In *ICALP*, pages 273–283, 1992.

[6] P. Buneman. A characterization of rigid circuit graphs. *Discrete Mathematics*, 9:205–212, 1974.

[7] A. Dress and M.A. Steel. Convex tree realizations of partitions. *Applied Mathematics Letters*, 5(3):3–6, 1992.

[8] W.M. Fitch. Toward finding the tree of maximum parsimony. In *The Eighth Interna- tional Conference on Numerical Taxonomy (Estabrook, G. F., ed.), San Francisco: W. H. Freeman and Company*, pages 189–220, 1975.

[9] W.M. Fitch. On the problem of discovering the most parsimonious tree. *American Naturalist*, 11:223–257, 1977.

[10] C. Johnson G. Estabrook and F. McMorris. A mathematical formulation for the analysis of cladistic character compatibility. *Math Bioscience*, 29, 1976.

[11] Martin Charles Golumbic, Haim Kaplan, and Ron Shamir. Graph sandwich problems. *J. Algorithms*, 19(3):449–473, 1995.

[12] D. Gusfield. Efficient algorithms for inferring evolutionary trees. *Networks*, 21:19–28, 1991.

[13] Dan Gusfield. Optimal, efficient reconstruction of root-unknown phylogenetic networks with constrained and structured recombination. *J. Comput. Syst. Sci.*, 70(3):381–398, 2005.

[14] Dan Gusfield. The multi-state perfect phylogeny problem with missing and removable data: Solutions via integer-programming and chordal graph theory. In *RECOMB*, pages 236–252, 2009.

[15] Dan Gusfield, Vikas Bansal, Vineet Bafna, and Yun S. Song. A decomposition theory for phylogenetic networks and incompatible characters. *Journal of Computational Biology*, 14(10):1247–1272, 2007.

[16] Dan Gusfield, Satish Eddhu, and Charles H. Langley. Efficient reconstruction of phylogenetic networks with constrained recombination. In *CSB*, pages 363–374, 2003.

[17] Dan Gusfield, Satish Eddhu, and Charles H. Langley. The fine structure of galls in phylogenetic networks. *INFORMS Journal on Computing*, 16(4):459–469, 2004.

[18] Dan Gusfield, Satish Eddhu, and Charles H. Langley. Optimal, efficient reconstruction of phylogenetic networks with constrained recombination. *J. Bioinformatics and Computational Biology*, 2(1):173–214, 2004.

[19] Dan Gusfield, Yelena Frid, and Daniel G. Brown 0001. Integer programming formulations and computations solving phylogenetic and population genetic problems with missing or genotypic data. In *COCOON*, pages 51–64, 2007.

[20] Michel Habib and Juraj Stacho. Unique perfect phylogeny is np-hard. To appear in CPM, 2011.

[21] Jotun Hein. Reconstructing evolution of sequences subject to recombination using parsimony. *Mathematical Biosciences*, 98(2):185–200, 1990.

[22] Katharina T. Huber, Vincent Moulton, and Mike Steel. Four characters suffice to convexly define a phylogenetic tree. *SIAM Journal on Discrete Mathematics*, 18:835–843, 2005.

[23] Sampath Kannan and Tandy Warnow. Inferring evolutionary history from dna sequences. *SIAM J. Comput.*, 23(4), 1994.

[24] Sampath Kannan and Tandy Warnow. A fast algorithm for the computation and enumeration of perfect phylogenies when the number of character states is fixed. In *SODA*, pages 595–603, 1995.

[25] Fumei Lam, Dan Gusfield, and Srinath Sridhar. Generalizing the four gamete condition and splits equivalence theorem: Perfect phylogeny on three state characters. In *WABI*, pages 206–219, 2009.

[26] Fred R. McMorris, Tandy Warnow, and Thomas Wimer. Triangulating vertex colored graphs. In *SODA*, pages 120–127, 1993.

[27] C.A Meacham. *Numerical Taxonomy*, volume G1 of *NATO ASI*, chapter Theoretical and computational considerations of the compatibility of qualitative taxonomic characters, pages 304–314. ed. Felsenstein J. Springer-Verlag, Berlin, 1983.

[28] C. Semple and M. Steel. *Phylogenetics*. Oxford University Press, 2003.

[29] Charles Semple and Mike Steel. Tree reconstruction from multi-state characters. In *Advances in Applied Mathematics 28*, pages 169–184, 2002.

[30] M. Steel. The complexity of reconstructing trees from qualitative characters and subtrees. *Journal of Classification*, 9:91–116, 1992.

[31] Lusheng Wang, Kaizhong Zhang, and Louxin Zhang. Perfect phylogenetic networks with recombination. *Journal of Computational Biology*, 8(1):69–78, 2001.